Facial Emotion Recognition with Vision Transformer (ViT)

Andrea Gurioli, Mario Sessa
Machine learning project, A.A. 2021/22
andrea.gurioli2@studio.unibo.it, mario.sessa@studio.unibo.it
0000984711 - 0000983529

Abstract: The project refers to some hot topics about Machine Learning: Transformers. Nowadays, Transformers obtained the State-of-Art in many tasks such as machine translations, sentiment analysis, code, and text generation. Generally, this kind of model works well in natural language processing tasks thanks to the complete correlations analysis on the whole text given by the concept of 'attention' described in [1]. There are few research papers on its usage in visual tasks such as facial emotion recognition. So, we will test it on hybrid dataset and provide a final application for the real-time inference of the model in the result.

1. Introduction

Nowadays, the facial emotion recognition (FER) task is still an ongoing research topic. Faces analysis indicates recognizing the angle and expression of a human being independently of the immersive environment it could be, and ambiguous emotions are the cornerstone of the problem. The focus of this project is to analyze how the vision transformer performs on this task, comparing our model with the State-of-Art models on hybrid datasets, taking into account the lack of inductive bias proper for Vision Transformer's (VIT) configuration introduced in [2] on small training datasets($< 14M$ images). Then, our next step is to adapt the system in a real use-case scenario, where the user can interact with an application and check his actual emotion displayed on his face in a real-time environment, exploiting a webcam input. If we want to obtain a valuable result, the input must be accurate and noiseless. Given this, the first part of the pipeline is composed of an SSD face detector, which returns in real-time the bounding box for a cropping purpose, then the cropped face becomes the input for the vision transformer classification. Generally, Transformers are data-hungry, and they need a considerable amount of data to be efficient as State-of-Art models. So, the challenge is to define a good FER model based on the VIT configuration with the capacity to detect facial emotions using a small amount of data. The documentation describes the pipeline and related procedures adopted in the project. We will explain: the data composition given by different datasets with high data variables, data integration to merge them into a unique dataset, data analysis which defines the features of each subset of data and defines some attributes and metadata to change for normalized samples, and data preprocessing for the data manipulation and augmentation to create a dataset split into three subsets with some features in common (image format, size, number of channels). In conclusion, we will discuss model configurations for face detection and cropping procedures and finetuned transformer for the Facial Emotion Recognition related to an evaluation analysis of results models.

2. Datasets

One of the first problem that we met, it's the availability of data. Many datasets are protected only for research uses and are not available completely for students. So, we will use only samples available on Kaggle or other open-source data platforms. Transformers need a good amount of samples to retrieve hidden patterns during training phase and the few data in our hands are not enought to satisfy this requirements. So, we have the plan to manipulate our small amount of samples to increase the size of the final datasets using data augmentation. The final dataset will have eight different classes integrated by three different subsets:

1. **FER-2013**: It contains approximately 40,000 facial RGB images of different expressions with a size restricted to 48×48, and the main labels can be split into seven types: $0 = Angry, 1 = Disgust, 2 = Fear, 3 = Happy, 4 = Sad, 5 = Surprise, 6 = Neutral$. The Disgust expression has the minimal number of 600 samples, while other labels have nearly 5,000 samples each.

2. **CK+**: The Extended Cohn-Kanade (CK+) dataset contains images extrapolated from 593 video sequences from 123 different subjects, ranging from 18 to 50 years of age with a variety of genders and heritage. Each video shows a facial shift from the neutral expression to a targeted peak expression, recorded at 30 frames per second (FPS) with a resolution of either 640x490 or 640x480 pixels. Unfortunately, we do not have all generated datasets, but we stored only 1000 images with high variance from a Kaggle repository.

3. **AffectNet**: It is a sizeable facial expression dataset with 60.000 images classified in eight categories (neutral, happy, angry, sad, fear, surprise, disgust, contempt) of facial expressions along with the intensity of valence and arousal.

Each dataset focuses on RGB channels for the colouring and has different sizes and image extensions entirely stored in Google Drive (the total amount of data is around 2 GB). So, we need to establish a standard format to manage them simultaneously. Finally, we have the final sections interested in the finetuning phase and training on a few models that can be saved locally on the drive and used in an external application for real-time classification on an ad-hoc application. More information about modelling is in the following sections.

3. Data Analysis

This section will talk about the features of each dataset described before and list any characteristics useful for data integration on AVFER dataset, which is the merged dataset, and preprocessing. Data Analysis uses some Python libraries to display and list images or their features like $glob, matplotlib, cv2, imageio, PIL, numpy, pandas$ and $matplotlib$.

3.1. FER-2013 Analysis

Fig. 1. Samples from FER-2013 dataset

FER-2013 is a dataset composed of 35.953 images in 7 classes (fear, disgust, sad, happy, neutral, surprise, angry). Images are in size 48x48 with a grey-scaled colours palette. The classes' variations and features distributions are helpful in the merging phase for other classes to obtain a good distribution and normalize the amount of data variation. According to the final classification, the contempt class was missed on this kind of dataset. Its composition already divides its original samples in train and validation set using different folder with the function of labelling. We can't use this structure because it is necessary to merge it with other datasets and proceed with the splitting in 3 different subfolder (train, test and val). Finally, we can analyze some features about the sample distribution separately between training and validation folders.

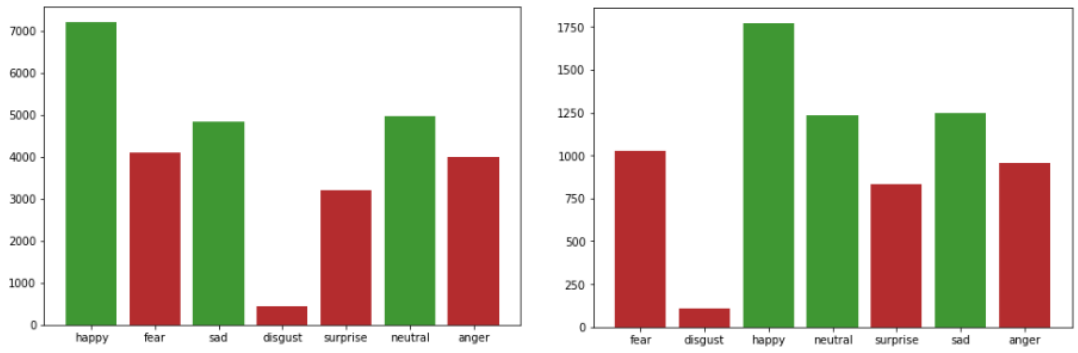

Fig. 2. Distribution of values in the FER-2013 dataset. On the left there is the training subset and on the right the validation one. Both plots display classes under (red) and up (green) value according to the mean quantity value.

FER-2013 does not have many samples for the disgust class. This observation guarantees an excellent motivation to merge it with other datasets and do the data augmentation phase. Furthermore, it is unbalanced, and we

need to manage it in data augmentation proportionally or data integration with some samples from other datasets to make a result set with the same number of samples for each class.

3.2. CK+48 Analysis

It is a small dataset composed of 981 images in seven classes (fear, disgust, sad, happy, neutral, surprise, angry). Images are in size 48x48 with a grey-scaled color palette. The classes' variations and feature distributions are helpful in the merging phase for other classes to obtain a good distribution and normalize the amount of data variation. Generally, images taken from video frames didn't have much variation, and the total number of elements is negligible compared to other datasets. Compared with the FER-2013, images are in frontal view with a clean pattern for facial expression.

Fig. 3. Samples from CK+48 Dataset

CK+48 contains less than 1000 images and didn't have a splitting in training and validation set as FER-2013. We will use it as part of the final training set for each dataset integration in our experiments.

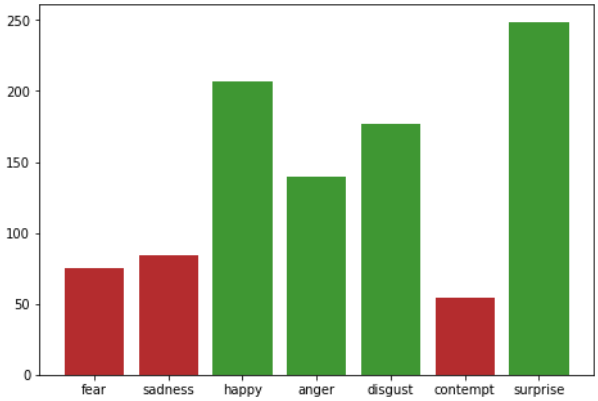

Fig. 4. Distribution of the CK+48 Dataset with the classification of classes under (red) and up (green) value according to the mean quantity.

Figure 4 displays the total samples from the CK+48 split by classes. According to the final classification, we don't have any samples about the disgust class, and contempt has fewer samples, pixel values are in the total range (0,255) for single channel. Images follow PNG formatting; we will convert it to the JPG to normalize images under a single format. The main difference between JPG and PNG is the compression algorithms. JPG uses a lossy compression algorithm that discards some of the image information to reduce the file's size. In comparison, PNG uses a lossless algorithm that keeps all the information; however, given the great amount of file that we will generate from data augmentation, we decided to have lower quality resolution images with reduced size than high

quality one because the low amount of data in PNG and the image noise applied on their convertion didn't have proportionally huge impact of the final dataset and it can add some noise to increase the resistance of the model.

3.3. AffectNet Analysis

AffectNet dataset [3] has samples of different sizes, high-quality images in grey-scale or coloured in RGB range. It has eight different classes (surprise, angry, sad, contempt, disgust, fear, neutral, and happy). As the FER-2013, there is a division between validation and training set but subfolders are almost balanced. We will exploit for the training, testing and validation splitting, given the validation set in the same subfolder for the final set and balance original training samples between the final testing and training subset for some experiments. In the last experiments, with hybrid validation set, we will use the original AffectNet validation set in the testing phase. As

Fig. 5. Samples from AffectNet Dataset

we can see from the Figure 5, AffectNet has some images colored and some in black and white with different face angle and backgrounds. During the data analysis, we make sure about the correct channels distribution for the whole original dataset. In other terms, we have verified that images have three channels for colored images. During analysis, empirically, we suppose for hypothesis that black and white samples were in RGB format too.

Fig. 6. In the first row, we have pixel values for each channel. In the second row, there are the same values visualized using the zero matrices given by *numpy.zeros* function in Python.

So, we made sure analyzing theirs pixels values in their channel distribution and we made as conclusion that for each of their pixels, we have the same value in the three channels. This aspect gives us the possibility to adapt simply FER-2013 and CK+48 to AffectNet samples using a channel conversion procedure and put the brightness

value, typically used for the pixel value in gray-scaled samples, for the three-channel of the pixels of every image and transform them into black and white images in RGB format. A problem encountered during the AffectNet analysis is given by the noise of the labels, indeed this dataset is very poorly annotated, and this kind of problem surely impacts the overall accuracy.

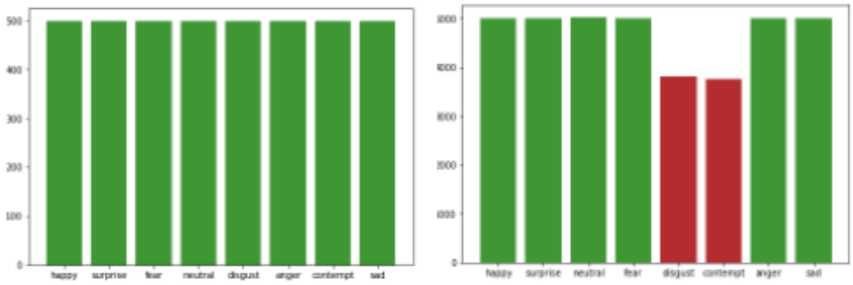

Fig. 7. Distribution of the AffectNet Dataset with original training (left) and vaidation (right) set. Plots shows also the classification of classes under (red) and up (green) value according to the mean quantity.

4. Data Integration

At the end of the Data Analysis of each data source, we merge it into the final dataset conventionally called AVFER (Aggregation for ViT on Facial Emotion Recognition). We have validation and testing sets balanced with the same number of samples for each class; meanwhile, the training set has a minimum amount of samples for every class of values but is not balanced. We will readjust the training set through data augmentation to reach a sufficient number of samples for each class. We will then eliminate the excessive generated images and create a dataset with the same number of samples for each class of the problem domain. It is correct to say that the amount of data for contempt and disgust are really low, even after the integration with available open-source data; we can try to increase the variance of pixel matrices without using oversamples techniques but only data augmentation, which increase the number of the minor classes on the training set to obtain a same value of samples distribution and make the dataset balanced with generated images with similar features.

5. Preprocessing

This section will discuss data manipulation to prepare them for the *DataLoader* object from *PyTorch* library. We will transform the PNG images into JPG format as defined in section 3.2; then, we convert images into three channels and transforms gray-scaled samples into RGB images. In conclusion, we carry out the data augmentation proportionated to each class's initial amount of images.

5.1. Data Manipulation

AVFER has some samples from CK+ as PNG, and their images have one channel (gray-scaled type). We will avoid artificial coloring and reduce the side effects of the fooling image. So we need to convert it to RGB, modifying only the number of channels and adjusting pixels values. In the gray-scale samples, each pixel has 1 byte (8 bits equals a value from 0 to 255, corresponding to the image's brightness value). We need to convert this pixel into three channels corresponding to 3 bytes equal to the red, green, and blue values during the transformation. In this conversion, each channel has the same value of brightness to maintain the gray tone of a colored pixel with no value perturbation. This operation uses the $OpenCV cvtColor$ function, which gets as parameters an image and a conversion type. For our objectives, we used the $cv2.COLOR\_BGR2RGB$ that change the color and channels from a gray-scale representation to RGB. Finally, we compared the converted image with an original black and white sample of AffectNet to check if the distribution of the values follows the same roles.

5.2. Data Augmentation

Data augmentation has 2 final objectives:

1. Balance the total amount of samples per class according to an arbitrary number chosen for the final dataset (20.000 samples per class).

2. Improve the total data of the training set and increase the variability of patterns for the training phase. This feature is significant for transformers' training because they need an huge amount of data to obtain good performances, even if we use finetuning.

Keras *ImageDataGenerator* class provides a quick and easy way to augment your images. It provides various augmentation techniques defined as a parameter on the object declaration. It could be used during the preprocessing phase or even during the training phase during the epochs iterations. Due to the unbalance class distribution, we decided to use it before the training phase. Expressly, we have set the following techniques:

1. **Zoom augmentation**: It either randomly zooms in on the image or zooms out of the image. ImageDataGenerator class takes in a float value for zooming in the *zoom_range* argument. We used 0.6 because we want a high variance between generated images and maintains a value smaller than 1 to zoom in on the image.

2. **Image rotation:** is one of the widely used augmentation techniques and allows the model to become invariant to the orientation of the object. The value is between 0 and 360. We rotate images until 10grades to adapt frontal images of FER-2013 and CK+48 on a similar face orientation to AffectNet faces and do not cripple already rotated images.

3. **Random Shifts**: It shifts horizontally and vertically the face of the image, our value is 0.2 for both types of shifting, the manipulation is equal to moving pixels in different quantities until to obtain a maximum of the 20% of the width and height images. This kind of manipulation can adapt images with some cropping on the face borders and always remains face parts like eyes, noses, lips, and eyebrows that are typically the most relevant elements for facial emotion classification.

4. **Brightness augmentation**: It randomly changes the brightness of the image. It is also a helpful augmentation technique because faces will not be under perfect lighting conditions most of the time. So, it becomes imperative to train our model on images under different lighting conditions. *ImageDataGenerator* defines a range to reduce or increase the brightness of an image; we have chosen a range of 0.2 to 0.6 to maintain a discernible pattern even in black and white images.

We carried out data augmentation proportionally on the initial amount of samples per class. We divided the final amount that we wanted to reach (20000 samples per class) by the initial amount; the result equals the number of augmented images we will generate. In other terms, we want to carry out an oversampling approach to balance each class with at least 20.000 samples and undersampling on augmented images if the total number of samples overflows the class limit.

6. Models

In this section, we will introduce the use of Vision Transformer Single-Step Detector model for, respectively, emotions classification and face cropping with their adaptations.

6.1. Face Detection and Cropping with Single-Shot Detector Network

The face detection phase is the first part of our pipeline. As this stage isn't part of the research, the module is treated as a black box. Its usefulness is in giving the vision transformer a proper input, cutting off the background. The one-shot feature map usage, used to predict the bounding box anchor's correction, is the main strength of the Single-Shot Detector (SSD). Multiple detectors, which are 3x3 convolutions with $k \times (C1+4)$ output channels. k is the number of anchors at that scale, C is the number of classes, the '1' is for the objectness score, and '4' is the anchors' correction. The detectors are applied to several activations in the backbone (ResNet in our case) and other activations created to extend the number of convolutional layers. [4] The detections are then filtered with the non-maximum suppression technique.

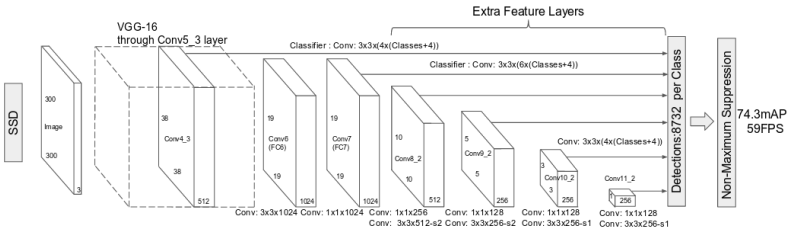

Fig. 8. Single-Shot Detector (SSD) with VGG-16 as a backbone

The resulting values of the model given by the OpenCV Framework are:

- The bounding box coordinates $[start_x, start_y, end_x, end_y]$
- The resulting confidence of the Face classification

When we filtered the overall detection results, results should be discarded with overall confidence lower than 0.5. The resulting bounding box is then resized as a square by taking the longest side of the resulting rectangle as the size of the final square. The Pytorch transform module then processes the cropped image by resizing it with a 224x224x3 shape as the VIT input dimension. Then the final step is to normalize it with the same values as the Vit finetuning phase: 0.5 of mean and 0.5 as standard deviation along all channels.

6.2. Vision Transformers

We exploit vision transformers by finetuning the pre-trained model on ImageNet, to classify eight human emotions: anger, contempt, disgust, fear, happiness, neutral, sadness, and surprise. Transformers are becoming the standard on NLP tasks. The core component in this kind of model is the attention mechanism, which can extract valuable features from the input with a standard query, key, and value structure, where the matrix multiplication between queries and keys, pulls the similarity between them. Then, the softmax function applied on the result is multiplicated on the value, obtaining our 'attention' mechanism. Our transformer architecture is based on a stack of eleven encoders preceded by a hybrid patch embedding architecture. The improvement is made by considering the lack of inductive bias problem. Vision Transformer has much less image-specific inductive bias than CNNs.

6.2.1. Vision Transformer Standard Structure

In the convolutional neural network, locality, two-dimensional neighborhood structure and translation equivariance are baked into each layer throughout the whole model. MLP layers are local and translationally equivariant in VIT configuration while the self-attention layers are global. The two-dimensional neighborhood structure is used very

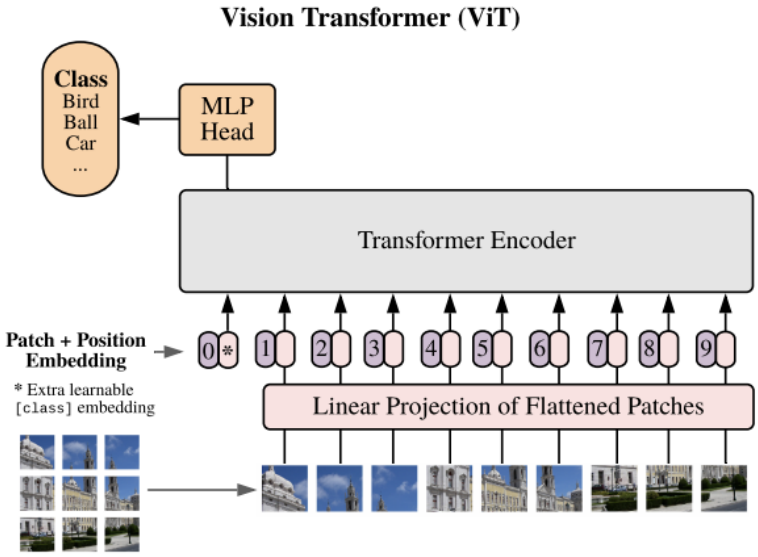

Fig. 9. Vision Transformer Standard Structure

sparingly: at the beginning of the model by cutting the image into patches and at finetuning time for adjusting the position embeddings for samples of different resolutions. Other than that, the position embeddings at initialization time carry no information about the patches' 2D positions, and all spatial relations between the patches have to be learned from scratch. As an alternative to raw image patches, the input sequence is formed from feature maps of a CNN (LeCun et al., 1989). The patch embedding projection is applied to patches extracted from a CNN feature map in this hybrid model. [2] The embedding of the feature map is obtained by adding the position embedding and a linear projection of the feature map obtained from the CNN's output. If we want to get valuable results in terms of accuracy for our task, it is preferable to adopt a configuration of the training phase starting from a pre-trained model. The focus passes in a finetuning process where the initial classification domain (ImageNet) is adapted in an eight-dimensional classification task. We obtain this result by changing the last head layer with an eight-neuron linear layer (one for each class). Then, the model follows finetuning with a Cross-Entropy Loss function and the stochastic gradient descent algorithm. For the final prediction phase, in order to obtain a probability distribution over the eight classes, a softmax function is exploited. In appendix C, we can see instances of our tool application implemented for Colab. A Javascript function exploits an application to capture the computer camera in real-time. Given that, the tool has an effortless use. Still, the computational delay increases as the frame data should be transmitted to the back-end for emotion prediction and retransmitted to the front-end for the final exhibition.

6.2.2. Vision Transformer with Sharpness-Aware Minimizer

Shaprness-Aware Minimizer (SAM) uses are motivated by the connections between the geometry of the loss landscape of deep neural networks and their generalization ability. It is also used for transformers to smooth the loss landscape and simultaneously minimize loss value as well as loss curvature thereby seeking parameters in neighborhoods having uniformly low loss value and generalize, following a more linear curvature on the loss values. This is indeed different from traditional SGD-based optimization that seeks parameters having low loss values on an individual basis and not on neighbors-relation. So, we can modify the Vision Transformer described in the previous section adapting optimizer on SAM.

This should generalize the minimization of the loss value but increment the training time of the model. [5]. The SAM optimization function can also tackle the 'noisy-label' problem present by obtaining a high degree of robustness to label noise datasets [6] which indeed is part of the problem taking into account the Affectnet dataset.

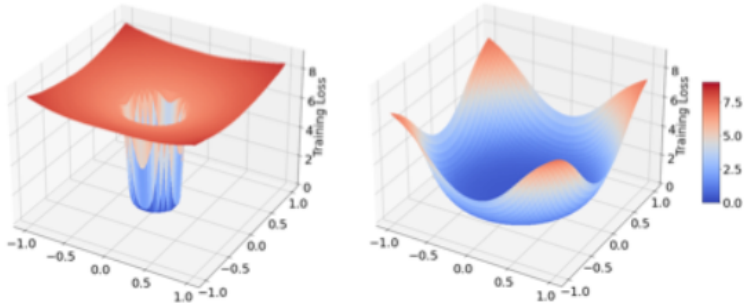

Fig. 10. Cross-entropy loss landscape on ViT (left) and the same smoothed landscape with the application of SAM during the training on ImageNet [5]

Furthermore, according with final considerations of [5], SAM works better with small datasets, mostly on Vision transformers like ViT-B/16. In conclusion, we have considered two assumptions: first, SAM incurs another round of forwarding and backward propagations to update neuron weights, which will lead to around 2x computational cost per update but get better performances on small datasets. Second, we notice that the effect of SAM diminishes as the training dataset becomes larger, but we can't have the not augmented data and obtain a balanced dataset at the same time; due to the low number of contempt and disgust samples.

6.3. Experiments

The proposed model starts from pre-trained parameters given by *timm* library available in Python. It allows us to download a pre-trained model with specific transformers configurations based on the number of neurons in the neural network. For each of their structure, it provides a random weighted-based version without a pre-training phase. Models implementation uses PyTorch components as the main framework. As described in the previous section, FER adaptation offers changes in the last linear layer configuration, reducing the total number of their neurons to eight units corresponding to the number of classes in our problem. During the preprocessing phase, we redefined the size of the images to adapt the dimension in 224x224 on three different channels (corresponding to the RGB channels); we described data Preprocessing more in detail in the previous section. Apart from the data augmentation phase, data follows some transformation during loading in the Dataset object. This process involves applying a Random Resize Crop technique, which crops part of the image and resizes it into the input format to focus aspects of images and defines hidden patterns only in these, increasing features classification more in detail. Then, we applied a Random Horizontal Flip and an image to tensor transformation to adapt the pixel matrix in a PyTorch tensor object. Finally, we normalized the input data and prepared samples for the training phase. The normalization phase applies a mean and a standard deviation to 0.5 for each channel. The best validation accuracy taken from the set of epochs during the training phase defines the final model weights set. Finetuning phase adapts the model parameters to the FER task using stochastic gradient descent or sharpness-aware minimizer adaptation with a cross-entropy loss function. The learning rate follows a scheduler that adjusts the initial value for every ten epochs by multiplying it to 0.1; we implemented it but, in practice, we can't follow it because the training time available by our environment have a time limit of 24 hours and we can't reach more than 20 epochs in one cicle of training, so we dump the weights of parameters and re-load them more times to continue the training in multiple sessions. Finally, we applied a simple momentum of 0.9 to increase the speed of training and a variable learning according to the optimizer chosen in the experiment. In conclusion, we used as the training environment Google Colab Pro with a P100 GPU and 16 GB of RAM available for a batch of 60 units. We carried out different experiments with different configurations on this environment according to compares different configuration for the Vision Transformer and for building a simple ResNet-18 model to compare performances with proposed configurations. [2]. They are:

1. **ResNet-18:** Follows the ResNet configuration with 18 deep layers described in [7] with 11 Milions of trainable parameters. We trained it to compare our transformers with small models to compare the trainability on our dataset on a model with a different structure and without the lack of inductive bias which is typical for the transformers. We trained ResNet on a gradual learning rate decrement and a momentum of 0.9 with SGD-based optimization on 25 total epochs.

2. **ViT-B/16/S:** It is our baseline. It follows the basic Vision Transformer with 16-units on patch embedding and final linear layer adaptation for FER. This model use SGD as optimizer with a fixed learning rate equals to 0.001 and a momentum of 0.9. We trained it on 25 epochs.

3. **ViT-B/16/SG:** Same configuration of ViT-B/16/SGD but with different learning rate starting at 0.01. We used a gradual learning rate decrement. We trained it on 25 epochs too.

4. **ViT-B/16/SAM:** It is equal to the baseline model ViT-B/16 with Sharpness-Aware Minimizer as optimizer. We used a gradual learning rate decrement and a momentum of 0.9. We trained it on a total of 25 epochs.

During the training phase, we saved histories of loss and accuracy for training and validation set. Every versions of model is trained on hybrid and augmented trainig set formed by 161k images equally distributed in 8 classes with AffectNet, FER-2013 and CK+48 samples and validated and tested only on an unused subset of AffectNet samples. We notice that noisy samples of AffectNet have a significant impact on performances. An example is reported by F. Ma, B. Sun and S. Li on their transformer adapted with Attention Selective Fusion in [8] that gets between 56% to 62% of the accuracy/robusness of their proposed model on the complete version of AffectNet.

7. Evaluation

7.1. Metrics Evaluation

The following table shows principal classification metrics results on the training phase. We tested models on 4000 different samples of AffectNet without data augmentation and with disgiunt relations with training and validation sets. The testing dataset is formed by 4000 samples equally distributed (500 samples per class).

Metrics	**Models tested on AffectNet**			
	ResNet-18	**ViT-B/16/S**	**ViT-B/16/SG**	**ViT-B/16/SAM**
8-Classes Accuracy	0.5005	0.5225	0.5242	0.5310
7-Classes Accuracy	0.5277	0.5489	0.5591	0.5694
Weigthed Avg. Precision	0.5090	0.5485	0.5404	0.5470
Weigthed Avg. Recall	0.5005	0.5225	0.5242	0.5310
Weigthed Avg. F1-Score	0.4943	0.5184	0.5169	0.5220
# Train. Params.	11.7M	86.5M	86.5M	86.5M
# FLOPS	1.8G	17.5G	17.5G	17.5G

Table 1. Testing Accuracy (with approximation to 8 and 7 classes), Weighted Average Precision, Recall and F1-Score on project models tested on AffectNet.

Due to lack of data for contempt class, we evaluated models on AffectNet even considering only the 7 less-augmented classes (original classes without contempt). Finally, for a more detail evaluation, we have written precision, recall and F1-Score distribution over classes and confusion matrix on the testing set for every tested model respectively in appendix A. and B.

7.2. Training Evaluation

We tried some different configuration about the use of SAM and the gradual learning rate on the ViT-B/16 configuration with the objective to find the best configuration, avoiding overfitting or underfitting and obtains acceptable performances using a small dataset.

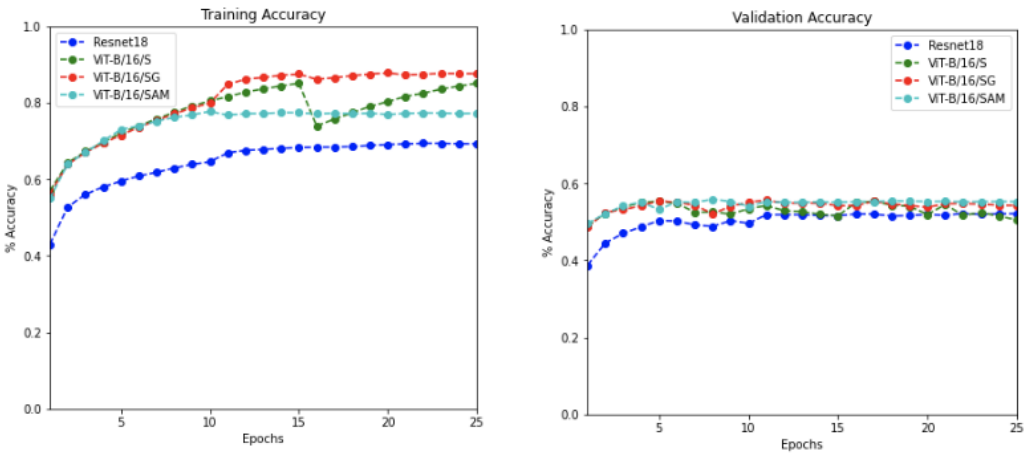

Fig. 11. The figure shows how training and evaluation accuracy changes over time for the models. The main differences are related to the gradual learning rate usage on the transformers for the learning rate that gives a gradual training accuracy increment in a regular curve, meanwhile the validation accuracy obtains an initial increment then reach the plateau around 55%, even for ResNet-18 model. ViT-B/16/S has an irregular curve due to the strong augmentation that introduces bias and impact on the regular improvement of the model, meanwhile other models are not affected by this kind of effects.

Hence, adapting the Vision Transformer on the gradual learning rate can make regular the training accuracy curves but performances in the validation accuracy remains similar.

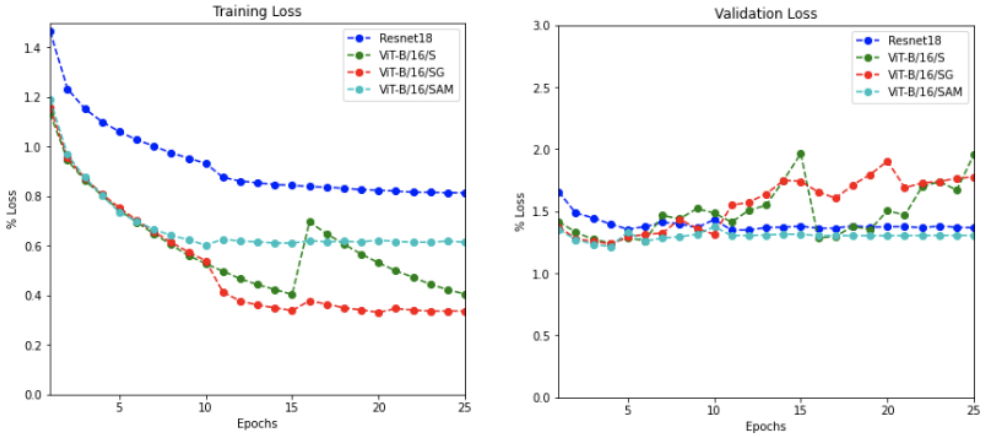

Fig. 12. The figure shows how training and validation loss changes over proposed Vision Transformers configurations and compares them with the ResNet-18 behavior. According to the plots, ResNet-18 mantains the high training loss over times compared to ViT configurations. ViT-B/16/S has an irregular curve due to the strong augmentation that introduces bias and impact on performances even for trainig accuracy. Validation loss of the SAM based model obtains a similar behavior of ResNet-18 due to the smoothing of the loss landscapes which performs well for small dataset as our case.

8. Conclusions

We have explored the direct application of Transformers to image recognition and test robustness on noisy datasets like AffectNet. Instead, we interpret an image as a sequence of patches and process it by a standard Transformer encoder as used in NLP. Related to [2] conclusions, we continued their works adapting a finetuned ViT with base configuration on FER on a small dataset. Our challenge was to test and obtain a model with the capability to recognize eight classes of emotions with the constraints of data availability for the FER task; we used only a subset of AffectNet, FER-2013, and CK+ for train and validate models. Furthermore, after testing and evaluating models, we tested them on a demo for a Real-time Facial Emotion Recognition combining our models with a Single-Shot Detector (SSD) for Face Detection and Face Cropping and labeling a face instantly. We compared the ViT-B/16 fine-tuned model with a ResNet model, also fine-tuned by us, both pretrained with ImageNet. We can say that the ViT-based model has similar performances than the classical ResNet model on small datasets, but it has many more parameters and requires more FLOPS for a standard forward process. In a use-case scenario where speed is a constraint, like in our tool of real time emotion recognition, models with a less number of FLOPS could speed up the whole prediction process.

References

1. *Attention is All You Need, Vaswani, Ashish and Shazeer, Noam and Parmar, Niki and Uszkoreit, Jakob and Jones, Llion and Gomez, Aidan N. and Kaiser, Lukasz and Polosukhin, Illia, 2017*
2. *ViT: An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, https://arxiv.org/abs/2010.11929*
3. *AffectNet: A Database for Facial Expression, Valence, and Arousal Computing in the Wild, Ali Mollahosseini, Behzad Hasani and Mohammad H. Mahoor, 2017*
4. *SSD: Single Shot MultiBox Detector, Wei Liu and Dragomir Anguelov and Dumitru Erhan and Christian Szegedy and Scott Reed and ChengYang Fu and Alexander C. Ber, 2016, https doi.org/10.1007%2F9783319464480_2*
5. *When Vision Transformers outperform ResNets without Pre-training or Strong Data Augmentation, Xiangning Chen, Cho-Jui Hsieh, Boqing Gong, March 2022, https://arxiv.org/pdf/2106.01548.pdf*
6. *Sharpeness-Aware minimization for Efficiently Improving Generalization, Pierre Foret, Ariel Kleiner, Hossein Mobahi, Behnam Neyshabur, Apr 2021*
7. *Deep Residual Learning for Image Recognition, K.He, Kaiming and Zhang, Xiangyu and Ren, Shaoqing and Sun, Jian, 2015*
8. *Facial Expression Recognition with Visual Transformers and Attentional Selective Fusion, Fuyan Ma, Bin Sun and Shutao Li, Fellow, Feb 2022*
9. *A Survey of Vision Transformers, Xang Liu, Yao Zhang, Yixin Wang, Feng Hou, Jin Yuan, Jiang Tian, Yang Zhang, Zhongchao Shi, Jianping Fan, Zhiqiang He, May 2022*

Appendices

A. Metrics Distribution on AffectNet Testing

Classes	Precision	Recall	F1-Score
Anger	0.4426	0.4780	0.4596
Contempt	0.5272	0.3100	0.3904
Disgust	0.5833	0.3360	0.4264
Fear	0.5846	0.5320	0.5571
Happy	0.5978	0.7460	0.6637
Neutral	0.3773	0.4520	0.4113
Sadness	0.4685	0.5800	0.5183
Surprise	0.4905	0.5700	0.5273

Classes	Precision	Recall	F1-Score
Anger	0.4264	0.6260	0.5073
Contempt	0.5348	0.3380	0.4142
Disgust	0.6644	0.3840	0.4867
Fear	0.5978	0.5380	0.5663
Happy	0.6144	0.7680	0.6827
Neutral	0.3850	0.6060	0.4709
Sadness	0.5464	0.5180	0.5318
Surprise	0.6185	0.4020	0.4873

Classes	Precision	Recall	F1-Score
Anger	0.4746	0.5240	0.4981
Contempt	0.5957	0.2800	0.3810
Disgust	0.6224	0.3560	0.4529
Fear	0.5955	0.5800	0.5876
Happy	0.6121	0.7920	0.6905
Neutral	0.3736	0.5380	0.4410
Sadness	0.5374	0.5460	0.5417
Surprise	0.5115	0.5780	0.5427

Classes	Precision	Recall	F1-Score
Anger	0.4622	0.5380	0.4972
Contempt	0.6037	0.2620	0.3654
Disgust	0.6401	0.3700	0.4689
Fear	0.6302	0.5760	0.6019
Happy	0.5959	0.8080	0.6859
Neutral	0.4012	0.5320	0.4574
Sadness	0.5323	0.5940	0.5614
Surprise	0.5108	0.5680	0.5379

Fig. 13. Tables of Precision, Recall and F1-Score for each classes. In order: ResNet-18 (left-up), ViT-B/16/S (right-up), ViT-B/16/SG (down-left) and ViT-B/16/SAM (down-right).

B. Confusion Matrices on AffectNet Testing

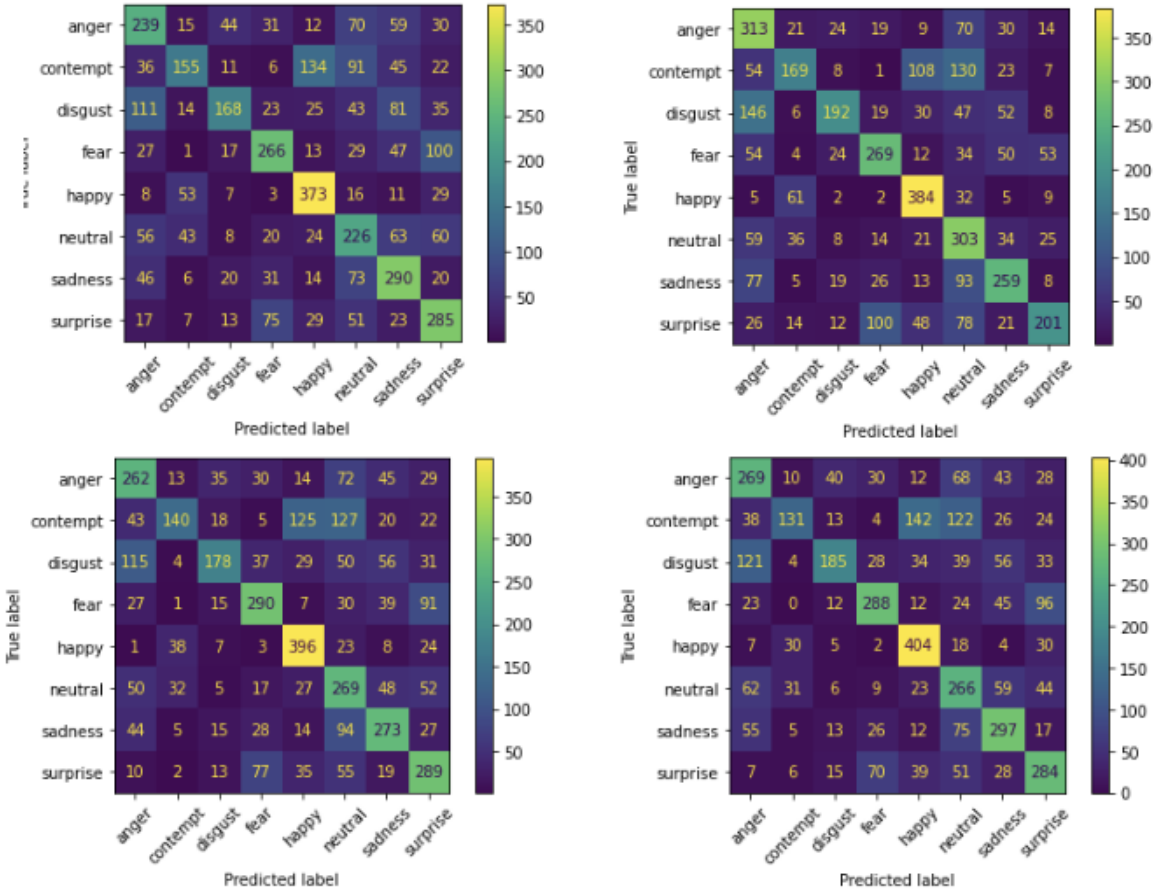

Fig. 14. Confusion Matrix on AffectNet testing. In order: ResNet-18 (left-up), ViT-B/16/S (right-up), ViT-B/16/SG (down-left) and ViT-B/16/SAM (down-right). We can see that performances are similar even for the ResNet configuration. Furthermore, the lack of contempt original samples can affects performances on the correct classification of this class.

C. Demo Application

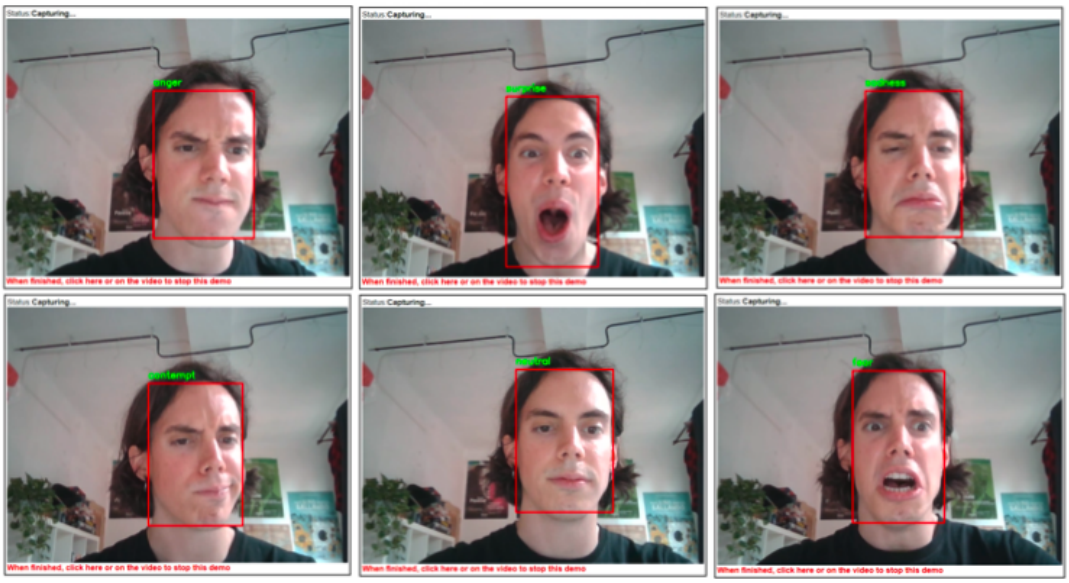

Fig. 15. Demo of Single-Step Detector implementation combined with ViT-B/16/SG. In order: anger, surprise, sad, contempt, neutral, and fear recognition examples. The Single-Shot Detector recognizes the face; hence it does face cropping, and the result face goes in input to the FER model for the real-time labeling.

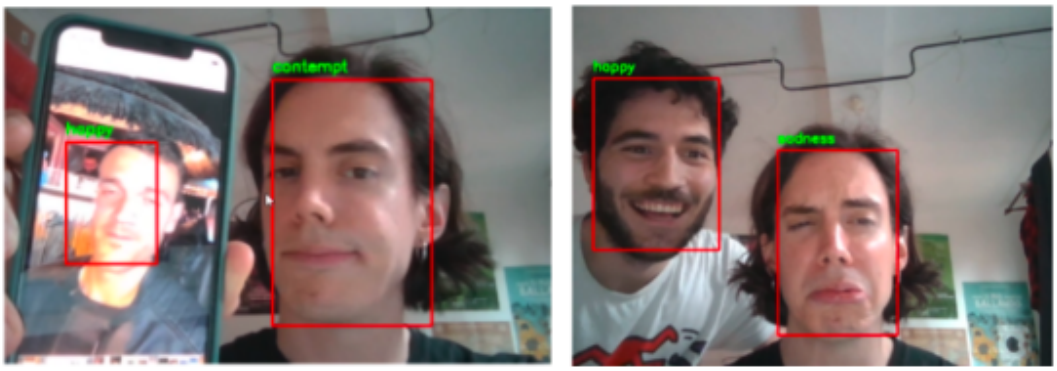

Fig. 16. The figure shows an empirical test on the Single-Shot Detector (SSD) for multiple face detection in the same frame with related labels for each clipping. On the left is an interaction with an image (phone photo) and a real face; on the right with two real faces.